DATA DRIVEN COMMUNICATION SYSTEM

FIELD OF THE INVENTION

The present invention relates generally to a communication system. More particularly, the present invention relates to a communication system that uses agents to enhance flexibility of the system.

BACKGROUND OF THE INVENTION

Most existing communication systems grew out of the public-switched telephone system, which was created as a point-to-point voice system. Based on the initial concepts and the technologies of the time, telephones and telephone numbers were identified with specific people or locations. In other words, traditionally a person is identified with a specific telephone in a person's home or office and that telephone is identified by a specific telephone number. That traditional mindset continues in the present day even though a person may have the use of multiple communications devices including landline telephones, cellular telephones and voicemail boxes. Accordingly, it is desirable to have a system that has increased flexibility and allows a person to receive a call using a desired device based, for example, on time of the day.

As technology advances, users are offered an increasingly varied number of features having greater complexity and sophistication. In conventional communication systems, such as telephone systems employing public branch exchanges (PBX) and central offices (CO), features are implemented in a call control system that uses a central resource manager. However, a serious drawback to this approach is that the call control system and the resource manager are very complex since they interact with or handle many aspects of the system, making modifications difficult to implement. In addition, modifications require highly specialized knowledge of the call control system and resource manager.

The difficulty of making such modifications results in an ineffective centralized system of requests for modifications or design changes. Typically, design change requests (DCR) received are centrally evaluated. Those requests believed to be of sufficiently high merit are planned to be implemented in a future release. This process typically involves many people or organizations and requires several months. In addition, a planned modification may not appear in the scheduled release if a subsequent DCR is given a higher priority.

Adding to the difficulty of the process is the highly specialized knowledge required. An experienced person who leaves an organization that maintains such centralized systems may be very difficult to replace and may affect the schedule of future releases or the number of modifications that can be made.

It is, therefore, desirable to provide a communication method and system that is flexible and eliminates the excessive complexity and difficulty in effecting system modifications.

SUMMARY OF THE INVENTION

It is an object of the present invention to obviate or mitigate at least one disadvantage of conventional communication systems. Generally, the present invention provides a data driven communication system that has three layers consisting of destination, people or network nodes, and devices. Software agents represent the entities of these layers and relationships between agents are defined by policies or chains of policies. These relationships are used to determine communication paths within the system as well as features associated with a particular communication. A database representation of the objects (agents and policies) is used to configure the system and receive updates to the system by an administrator. A graphical user interface can be used to enter data into the database and facilitates an intuitive understanding of the nature of the relationships involved.

In a first aspect, the present invention provides a method for establishing a communication path in a data-driven communication system. The method consists of defining a first relationship between a first layer agent and a second layer agent, and defining a second relationship between the second layer agent and a third layer agent. Next, a first communication link, between the first layer agent and the second layer agent, is established according to rules of the first relationship. A second communication link is then established, between the second layer agent and the third layer agent, according to rules of the second relationship. The first and second communication links establish a communication path.

In a presently preferred embodiment, the first layer agent is a destination agent, the second layer agent is a node agent, and the third layer agent is a device agent. The first and second relationship are defined by associating at least one policy, policy chain or branched policy chain with each of the relationships to define its rules. The communication path is established by providing system parameters, such as system time and system date, to the

policies.

In a further aspect of the present invention, there is provided a data-driven communication system. The communication system includes a first layer agent, a second layer agent and a third layer agent. A first relationship, between the first layer agent and the second layer agent, is used to establish a communication link between them in response to data provided to the first layer agent. A second relationship, between the second layer agent and the third layer agent, is used to establish a second communication link between the second and third layer agents, in response to data provided by the second layer agent.

Again, in a presently preferred embodiment, the first layer agent is a device agent, the second layer agent is a node agent, and the third layer agent is a destination agent, and policies, policy chains or branched policy chains define the first and second relationships. System features can be provided for modifying the communication path. The system features can include in-call features, data modifying features, and advanced programmable system features.

It is presently preferred that the agents and policies be implemented as objects, through an acceptable object-oriented programming language. Typically, the objects are organized as records in tables in a database. Such a database permits the communication system to by configured at startup, or reconfigured as desired. A graphical user interface, displaying icons representing agents and policies, facilitates modification of the database.

Other aspects and features of the present invention will become apparent to those ordinarily skilled in the art upon review of the following description of specific embodiments of the invention in conjunction with the accompanying figures.

BRIEF DESCRIPTION OF THE DRAWINGS

Embodiments of the present invention will now be described, by way of example only, with reference to the attached figures, wherein:

Figure 1 illustrates schematically a conventional communication system;

Figure 2 illustrates the software architecture of a communication server according to an embodiment of the present invention.;

Figure 3 illustrates the concept of a relationship between two agents;

Figure 4 illustrates the concept of a policy;

Figure 5 illustrates an example of the relationship between two agents;

Figures 6A and **6B** illustrate examples of different types of policies;

Figure 7 illustrates the relationship between a database corresponding to the agents of the present invention;

Figure 8 illustrates the relationship between tables and objects (specifically, agents and policies);

Figure 9 illustrates the setup of the system by a user using a data entry station;

Figure 10 illustrates the use of the database server to set up the communications server;

Figure 11 illustrates how the system is updated;

Figure 12 illustrates an initial layout of objects for a graphical user interface for configuring the system;

Figure 13 illustrates the objects of **Figure 12** with relationships defined between agents;

Figure 14 illustrates an example of a configuration for a particular destination (222);

Figures 15A to **15C** illustrate the association of a person with a device;

Figure 16 illustrates an example of a configuration for two destinations directed to a person's node;

Figure 17 illustrates schematically two half calls forming a call;

Figure 18 illustrates an example of line selection for an originating half call;

Figures 19A to **19C** illustrate three types of features of the embodiment of the present invention;

Figures 20A to **20C** illustrate examples of each type of feature;

Figure 21 illustrates the use of policies by a feature;

Figure 22 illustrates the use of a trigger table to determine which feature is selected; and

Figure 23 illustrates the use of a policy to determine which APS feature is triggered.

DETAILED DESCRIPTION

A conventional communication system is illustrated in **Figure 1** and includes a communication server **100**, a database server **200**, a network server **300** and a phone server **400** interconnected by a local area network (LAN) or wide area network (WAN) **500**. Telephones **410**, and other analogous devices, are connected to the phone server **400**. The

network server **300** is typically connected to a public switched telephone network (PSTN) **310** and a WAN **320**.

Generally, the present invention provides a method and system for data driven communication. In particular, the invention relates to the use of software entities and policies in the software architecture of communication server **100**. Software constructs or entities are used to determine communication paths for communication calls. As exemplified by the embodiments of the invention discussed below, these entities can be organized into layers. By layers, we mean groupings of similar entities. Associated policies are used to determine the attributes or properties of relationships between the entities and communications following the communication paths. The system is "data driven" in the sense that data associated with a communication or call, such as the number being called, the time of day, the day of the week and the originator of the call can determine the communication path of the call and the associated attributes of the call.

The invention can be implemented in software using numerous approaches. The following example embodiment of the present invention uses an object oriented approach to system design and programming. As is known in the art of software development, under an object oriented approach, classes are established or defined and objects are instances of a particular class consisting of both data and procedures or methods to manipulate the data. According to the exemplary embodiment which embraces an object oriented approach, agents, policies, features and other software constructs are implemented as objects.

As used herein, "agent" refers to a software entity used to represent a corresponding real-world counterpart. Agents will be discussed in greater detail below. By "policy", we mean a software construct or object that is used to define relationships between agents. These relationships include rules for determining communication links and their properties. In particular, the relationship between two agents is defined by the policies associated with the communication link between the two agents.

Referring to **Figure 2**, the software architecture of an embodiment the present invention has three layers, namely, destinations, people or network nodes, and devices. Entities of each layer are represented by agents **110**. Specifically, as illustrated in **Figure 17**, there are destination agents **112**, person or node agents **114**, and device agents **116**. It should, however, be noted that the present invention is not restricted to three layers and that fewer or more layers are also possible in alternative systems. The three layer system described herein

is especially applicable to a PBX replacement. However, a system can be designed to have two layers, or more than three, if desired. There can also be sub-layers within layers. Furthermore, entities are software constructs used by the system to establish communication paths. In the present example, it is convenient to associate entities with agents that correspond to real world counterparts. This is not, however, a necessary requirement of the architecture and the invention includes the possibility of other types of entities that may be entirely internal to the system.

Referring to **Figure 3**, relationships **130** between agents **110** are defined by policies **120**. A series of policies **120** forms a policy chain **126**. In a communication, a communication path is established by the use of these agents. Specifically, a communication path from a first agent to a second agent consists of the communication link between the first and second agents. If there are one or more intermediate agents then the communication path consists of the links between the first link to the one or more intermediate agents, any links between the one or more intermediate agents and the between the one or more intermediate agents to the second agent.. The communication path established from the first agent to the second agent defines the attributes of the communication.

It is possible for two agents to be directly connected to each other. Indeed, this corresponds to the default case where "nothing special" happens. For example, a person node may be directly connected to a device if that person uses only one communications device, such as a landline office telephone with no voicemail, or other added features. In such a case, no additional processing logic is required and agents can be directly interconnected with no intervening policies. This direct connection is typical of conventional call control systems in which a phone number is tied to a device. Any changes to the system, for example to add increased functionality such as voicemail, conferencing, etc., require working around this inherent coupling between a phone number and a device, and patching or adding to the system in a piecemeal fashion.

By contrast, the architecture of a system according to the present invention does not assume that a phone number is always associated with a person, or that a person is always associated with a particular device. Of course, such a direct association can be easily implemented in the present invention.

The present invention facilitates the addition of enhanced functionality by the use of agents to represent destinations, nodes, and devices. Simply adding policies **120** to the policy

chain **126** that defines the relationship **130** between agents **110** increases functionality. Changes in functionality can be simply effected by suitably modifying the corresponding policies **120**. If the system is implemented using object-oriented programming, then changes to an object class can implement a consistent global set of changes.

In addition, a system according to the present invention is easy for developers and system administrators to understand. Once the basic architecture of the system is understood, a straightforward review of the relationships between agents will make it clear what functions and features are invoked in a communication session.

Each policy **120** consists of a policy identification (ID) **122** and a specific instance of a policy **124** as illustrated in **Figure 4**. Policy ID **122** points to a specific policy **124** by employing a pointer **128**. Each specific policy **124** can point to a further policy **120** in the policy chain **126** through use of a pointer **129** pointing to the policy ID **122** of the further policy **120**.

A specific example of policies **120** defining the relationship **130** between two agents **110** is illustrated in **Figure 5**. In the example, "222" is a destination and could, for example, represent a telephone number of a person, Fred. A destination agent **112A** representing destination "222" is linked to a node agent **114A** representing Fred through a policy chain of policies **120A** and **120B**. Policy **120A** consists of policy ID **122A**, which has type equal to Day of Week and ID (i.e. the address of a specific Day of Week policy) equal to X. In other words, policy ID **122A** points to specific Day of Week policy X **124A**. As suggested by its policy type, specific policy X **124A** points, via pointer **129A**, to different objects depending on the day of the week. In the example, policy **120B** corresponds with the day of the week of the call. However, if the call occurred on a different day of the week then another policy such as policy **120C** could have been selected.

Policy **120B** consists of policy ID **122B** which identifies the policy type as "node" and the specific policy as node policy Y **124B**. Specific policy Y **124B** points to node agent **114A** representing Fred. The meaning of the relationship between the destination agent **112A** for destination "222" and the node agent **114A** representing Fred is that if "222" is dialed on certain days of the week then the call will be connected to Fred. Of course, Fred is a person and cannot be contacted directly. Further agents will link the node agent **114A** representing Fred to a suitable device. This will be illustrated in detail later.

Figure 6 illustrates different types of policies. **Table 1** in **Figure 6** lists a few

representative examples of policy types but many others could exist. A particular policy type included in **Table 1** is the group policy. Group policies are useful for logically organizing individuals. For example, individuals who are designated to respond to user requests about technical matters could be the members of a help desk group.

Policies **120** represent relationships **130** between agents **110** at different layers, but policies can also exist within policies. If we think of policies as rules for branching between agents of different layers then policies inside policies add refinements or sub-rules to the branching rules work.

In the example of **Table 2** of **Figure 6**, a policy inside a policy mechanism, i.e. a selection policy within the group policy, is used to select an individual from that group to respond to the call at hand. This would clearly be useful in distributing work to the members of our help desk group of in our earlier example. Different selection policies use different strategies for passing on the call. A terminal selection policy starts at the head of the list and passes the call to the first available person implementing a hierarchy or priority system of distributing calls. A circular selection more evenly distributes calls than a terminal selection policy by beginning to hunt where the last hunt left off. A broadcast policy passes the communication to everyone at the same time, for example, when a voice mail message is forwarded to everyone in the group. A longest idle selection policy begins by selecting the group member who has been idle the longest.

Another example of a policy inside a policy is a destination policy within a node policy. For instance, different telephone numbers could result in a communication path ending up at the same node but the different calls may continue along different paths out of the node. Accordingly, the node policy points to a different place, i.e. chooses a different policy path, depending on the desired destination. This provides another example of the flexibility inherent in the present system that is unavailable in conventional call control systems. To take a specific example, suppose that the manager of a help group has an assistant who receives calls to the manager to help screen them. On a particular day, the help group is short staffed and the manager agrees to take help desk calls, but still wants the assistant to screen regular calls. In a conventional system, the manager's number would be entered into the help desk group with the result that calls would first go through the assistant. Of course, inline code could be used to make an exception to the usual call processing but this is awkward and exemplary of a patchwork approach typical of conventional systems. By

contrast, a system that uses the present invention can easily implement such an arrangement by noting that the destination of the call is the help desk and distinguishing this call from calls to the manager. Accordingly, calls to the help desk routed to the manager go directly to the manager's phone whereas calls to the manager go to the assistant. The ease with which the system accommodates such changes is a direct consequence of the more powerful architecture of the system, using the layered agent approach.

A key aspect of the system is the use of a database to store information about the system. Entries within tables of the database represent objects, including agents and policies, within the system. The database is used to configure the communication server upon start up. Changes to the objects can be entered into the database, preferably using a graphical user interface. Additionally, any changes made directly to the system will also be recorded in the database so that no information is lost the next time the system starts up or is reconfigured.

Figure 7 illustrates tables **144**, **146** and **148** which correspond to the destination, the node layer, and the device layer, respectively. Generally, entries or records **142** in a table correspond to objects. Thus in **Figure 7**, the records **142** of the destination table **144** correspond to destination agents **112**, the records **142** of the node table **146** correspond to node agents **114** and the records **142** of the device table **148** correspond to device agents **116**. Similarly, the records **142** of different policy tables **150** correspond to the policies **120** of the system.

In addition, the interrelationships between records **142** in the database correspond to those between objects. This is shown in the example of **Figure 8** in which records **142** in the destination table **144A** point to records **142** in a Time of Day policy table **150A** or a Day of Week policy table **150B**. This is directly analogous to the example of **Figure 5** in which destination agent **112A** for destination "222" points to Day of Week policy **120A**. Recall that Day of Week policy **120A** consists of policy ID **122A** and specific policy X **124A**. Similarly, a record **142** in destination table **144A** specifies a policy type (indicating the type of table) and a policy ID (indicating which record in the table). From policy tables **150A, 150B**, further policies **150** can be specified until a node table **146** is reached. Then additional policy tables **150** are possible until the device table **148** is reached.

A user **220**, or system administrator, can make changes to the system by entering these changes in the database. The user can enter the changes by means of a data entry station **210** as illustrated in **Figure 9**. The information **230A** entered into the data entry station **210** is

used to update the database within the database server **200** as illustrated by arrow **230B**. When the communication server **100** is next brought up or initiated by a user **220**, it receives the data **230C** from the database server **200** and creates the appropriate objects (agents and policies). See, for example, **Figure 10**.

Figure 11 illustrates that a user **220** can update the system using either the data entry station **210** or a device such as a telephone **410** connected to a phone server **400**. The changes are entered into the database in database server **200** and then updated in the communications server **100**. Note, however, a buffer **160** can be used to hold changes to avoid disrupting existing communications.

Figure 12 illustrates an example user interface for entering changes to the database at the database entry station **210**. Icons corresponding to destinations, persons or network nodes, devices, and policies are available for manipulation in a default configuration. Specifically, there are icons **212** representing destinations, icons **214** representing policies, icons **216** representing nodes, icons **218** representing additional policies and icons **219** representing devices. Unnecessary icons can be removed or additional icons can be added. These icons are organized in rows, somewhat analogous to the layers of the different agents. In order to establish a desired configuration the user uses the interface to connect these icons defining possible communication paths (by the choice of icons representing agents) and defining properties associated with those communication paths (by the choice of policies).

An example configuration is shown in **Figure 13** which illustrates a web of destination, policy node and device connections through the layers. Note that policies can connect to other policies forming a policy chain.. It is also possible to have a direct connection without an intermediary policy if no processing logic is required, for example, if a person always wants to be reached by one particular phone such as a cellular telephone.

A more specific example is given in **Figure 14** in which a particular phone number (i.e. destination) "222" needs to be routed differently depending on the day of the week. In the example, the phone number of the help desk is "222". In this example, the number "222", represented by destination agent **112**, is the phone number of a help desk which is staffed by two groups. Group 1, represented by policy **120D**, is responsible for responding during the week and Group 2, represented by policy **120B**, works weekends. John, represented by node agent **114C**, works seven days a week and thus belongs to both groups. John works in the office during the week but prefers to work from home during the weekend. **Figure 14**

illustrates how such a configuration is easily arranged using the objects of the present invention.

The destination agent **112** corresponding with a call having destination "222" points to Day of Week policy **120** which in turn points to Group 1 policy **120C** on Monday to Friday and Group 2 policy **120D** on Saturday and Sunday. Group 1 has a default terminal hunt type starting with Fred, represented by node agent **114A**. Accordingly, Fred always gets a help desk call during the week if he is available, otherwise the call passes onto George, represented by node agent **114B**, and then to John at node **114C**. This terminal hunt strategy is used to implement a hierarchy or priority scheme in which Fred, at the head of the list, is responsible for most calls, with George and John acting to handle overflow or backup. No intermediate policy is required for Fred and Phone 1, represented by device agent **116A**, or George and Phone 2, represented by device agent **116B**, since Fred and George are available if and only if they are by their office telephones. For John, however, the Day of Week policy **120** is used to determine whether his office phone (Phone 3, represented by device agent **114C**) or his home office phone (external number "555-5555" represented by an external number agent **118A**) is the appropriate device to reach him, depending on the day of the week.

Calls during the weekend are routed to Group 2 by Day of Week policy **120**. In this example Group 2 uses a circular hunt type. Thus if the last call was received by Nancy, represented by node agent **114D**, then the system will try Mary, represented by node agent **114E**, and then John before passing another call to Nancy. In contrast to the terminal hunt strategy, this circular hunt strategy is used to equally distribute the work load of responding to calls.

The example of **Figure 14** illustrates an important aspect of the invention. Starting from a first agent, the relationships (defined by policies) associated with that first agent (e.g. destination agent **112**) are used to determine a communication path and the second agent (e.g. external number agent **118A**) through an intermediate agent (e.g. node agent **114C** for John). Note that the policies encountered starting with destination agent **112** determines the communication path and that the second agent is not predetermined. Instead the communication path and the second agent are determined, in this example, by the day of the week and the state of the system (e.g. whether Fred and George are available).

The calls to John at his external number illustrate a further aspect of the invention.

John wants his calls to go to his home at external number "555-5555". The system administrator wants to ensure that the best route for calling is used, based on system policies. So instead of having a trunk hooked up to John directly, the system finds an appropriate line. Referring to **Figures 15A** to **15C**, when the node agent **114C** for John has selected an external number the destination agent **112B**, representing that number, is asked to provide a device for this call. Intermediate policies and nodes are used to provide a connection that is consistent with the priorities and preferences of the system. A device, represented by device agent **116G**, is then selected and John's agent **114C** is notified of the selection and, for the duration of the call, John is associated with this device.

A second example of a configuration is provided in **Figure 16** which shows the paths for two destinations "333" and "444" represented by destination nodes **112B** and **112C**, respectively. Both of these destinations point to a node **114F** representing Joe. Joe's node **114F** points to two group policies, Group X **120D** and Group Y **120E** with their internal hunt policies set to terminal continuous and broadcast, respectively. Calls corresponding to destination "333" go to Group X **120D** whereas calls corresponding to destination "444" go to Group Y **120E**. Group X **120D** contains Phones 1 and 2, represented by device agents **116A** and **116B**, and Group Y **120E** contains Phones 2 and 3, represented by device agents **116B** and **116C**. Accordingly, if "333" is called, Phone 1 will ring followed by Phone 2 if there is no answer at Phone 1. If "444" is called then both Phones 2 and 3 will ring.

To establish a call, a minimum of six agents **110** are generally involved. **Figure 17** shows that a call is split into two sides or half calls **180**. Each half call **180** has at least one device agent **116**, one node agent **114** and one destination agent **112**. There is also a communication record **170** associated with each half call **180** that tracks all information associated with that half call **180** and is discarded after the half call terminates. This is a typical half call model used in many call control systems.

When a user originates a call using a device, a policy determines which node and destination to use. Specifically, the policy determines which path should be chosen. The example of **Figure 18** shows two destination agents **112** and two node agents **114** connected to one device agent **116**. When this device originates, it has up to four possible paths to follow. A policy **120** set in the device agent **116** will determine which path to take. The policy can be set statically, i.e. programmed into the database, or dynamically, i.e. by having the user choose which path is taken at the moment the call is made. In addition, policy can be

used to determine whether a static policy or a dynamic policy is chosen. In **Figure 18**, an example static policy **120D** always selects the path corresponding with Node Agent 1 and Destination Agent 2.

Features are also very important elements in a communication system. **Figures 19A** to **19C** show three types of features. Referring to Figure **19A**, type 1 features relate to the manipulation or selection of a destination or connection **182** between two half calls **180**. They are in-call call features such as swap and conference. As illustrated in **Figure 2**, type 2 features relate to setting up or updating data **162** in or associated with specific agents **110**. Examples of these type 2 data modifying features include "Do Not Disturb" and "Restriction List". Type 3 features are advanced programmable system (APS) features that give feedback to a user **220** who can then make selections. Intelligent tones or the use of interactive voice response (IVR) technology to replace a tone with a message are examples. Features of the system can also be elements of the system that are implemented as objects that can modify the communication path, its properties and the second agent at which the communication path terminates.

Further examples of each type of feature are provided in **Figure 20**. The examples of **Figure 20** are not exhaustive and additional features are possible. The focus of the present examples is on the use of agents and polices by a system to increase functionality and flexibility. Greater sophistication can, of course, be introduced by adding more intelligence and functionality to the agents or policies. For example, APS features can be used to get more information from callers using features in order to give allow agents to do more. More specifically, a caller to a service provider may be asked to provide an identifier such as an account number through IVR. Depending on the account number, the call may be routed to different attendants or services. For example, if the service provider is an insurance company, then the account number will identify the name of the customer and that will be used to direct the call to the person responsible for that person's account. As another example, the service provider may be a credit card company offering different levels of service (silver, gold, platinum, etc.) to different card holders. The caller's account number can then be used to route the caller to an appropriate queue or otherwise provide an appropriate level of service.

Features can also use policies. The data that a feature uses can be chosen based on policy. This is illustrated in **Figure 21**. When a feature **190** retrieves the data **196** it needs, it can do this through a set of policies **120** which then determine which data **196** is used. An

example of this is Time of Day Call Forwarding, where the destination that the call forward uses is determined by a Time of Day policy. A Time of Day policy may, for example, forward calls to a help desk during office hours but invite the caller to leave a voicemail message outside of office hours.

Features **190** are triggered or invoked in a particular agent **110** through an event occurring in a particular state. **Figure 22** shows how policy can be used to flexibly determine which feature is triggered. When an event Z is received by an agent **110**, there is an associated communication record **170** which contains the call state. The agent **110** then consults its trigger table **198**. The trigger table **198** contains a list of states and events, and corresponding policy IDs. The policy IDs point to policies **120** that eventually point to a particular feature **190**. In this way, features **190A**, **190B** can have the same state and event trigger, but through a policy, one can have priority over the other. This can also be based on other policies like Time of Day or Calling ID, etc. This means that we have a new policy type of feature. The ID would point to which feature.

APS features are triggered in a similar way. **Figure 23** shows how a policy can be used to flexibly determine which APS feature is triggered. When a specific type of event, i.e. tone T, is given by an agent **110** through in-line code then a trigger table **198** is consulted. This trigger table **198** contains a list of associated parameters, specifically, tones and reasons and corresponding policy IDs. The policy IDs point to policies **120** which eventually point to APS features. For example when giving a dial tone to start a call, an APS feature may be triggered to select a line. Trigger tables also reside in the database and can be uniquely assigned to an agent.

New policies can be added very simply in this system. The architecture provides the building blocks to create any sort of communication system needed by customers, based on their needs and organization.

The above-described embodiments of the present invention are intended to be examples only. Alterations, modifications and variations may be effected to the particular embodiments by those of skill in the art without departing from the scope of the invention, which is defined solely by the claims appended hereto.